Calidad y Preprocesamiento de Datos

MCIC VÍCTOR MANUEL CORZA VARGAS

MC FERNANDO AVITÚA VARELA

Tema 1 Integración de datos heterogéneos

1.1 Sistemas de bases de datos heterogéneos

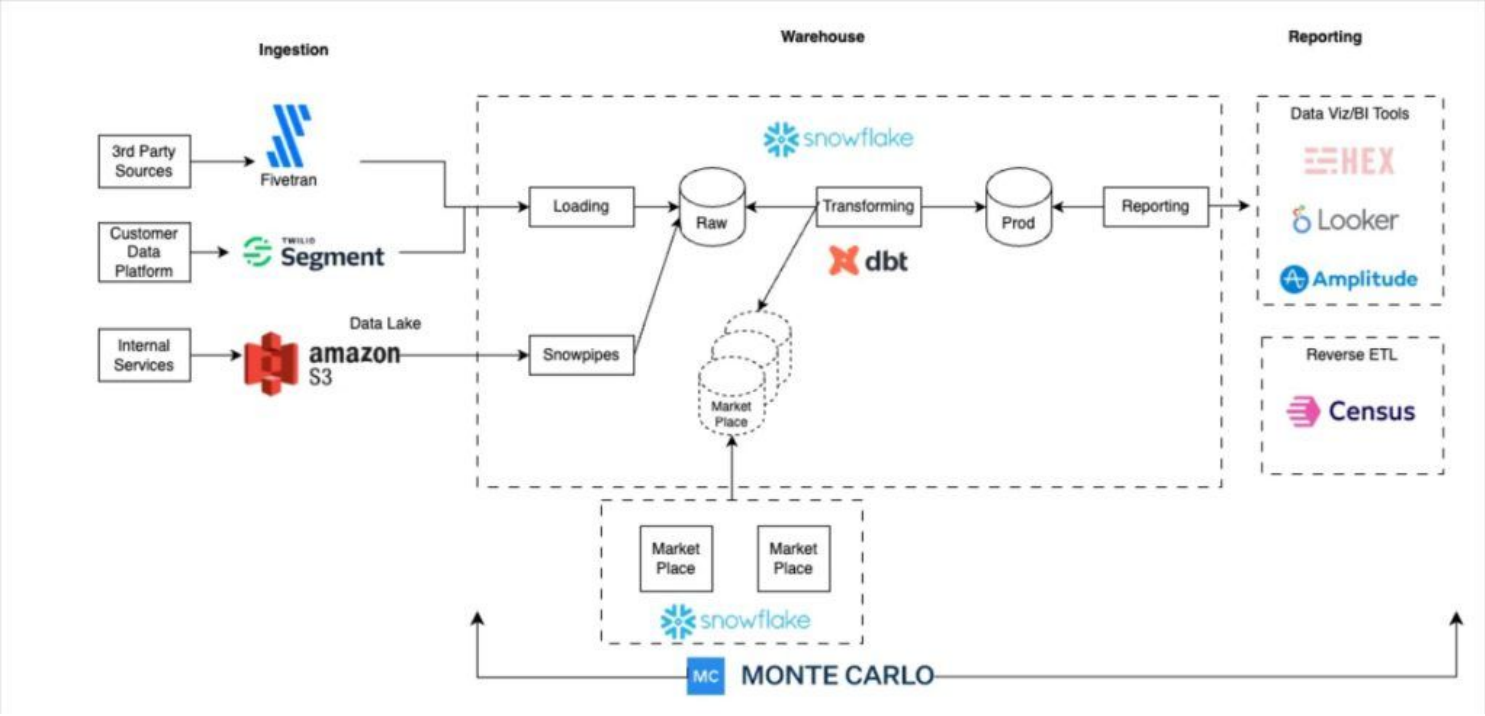
Objetivos:

- Comprender los conceptos fundamentales de la **integración de datos heterogéneos**.
- Identificar los desafíos asociados a la **heterogeneidad semántica** y **sintáctica** en sistemas de bases de datos.
- Analizar **casos prácticos** donde se presentan problemas de integración de datos.

Conceptos fundamentales de la **integración de datos heterogéneos**

Definición de Integración de Datos Heterogéneos:

Proceso de combinar datos provenientes de diversas fuentes con diferentes formatos y estructuras para obtener una visión unificada.



Swimply

Conceptos fundamentales de la **integración de datos heterogéneos**

Heterogeneidad en Sistemas de Bases de Datos:

- **Heterogeneidad Semántica:**
  - Diferencias en el significado e interpretación de los datos entre sistemas.
  - Es difícil de detectar con pruebas (errores silenciosos), se requiere conocimiento de negocio
  - **Ejemplos:**
    - El término "cliente" puede referirse a una persona física en una base de datos y a una empresa en otra.
    - Columna “Última compra” en base A es < 30 días, en base B es < 90 días
    - Ingreso bruto vs neto

Conceptos fundamentales de la **integración de datos heterogéneos**

Heterogeneidad en Sistemas de Bases de Datos:

- **Heterogeneidad Sintáctica:**
  - Variaciones en los formatos y lenguajes utilizados por los sistemas de datos.
  - Heterogeneidad es detectable por pruebas
  - **Ejemplos:**
    - Un sistema puede almacenar datos en formato XML, mientras que otro utiliza JSON.
    - Fecha *YYYY-MM-DD* vs *DD/MM/YYYY*
    - *null* a *None* a *“”*

Conceptos fundamentales de la **integración de datos heterogéneos**

Impacto de la Heterogeneidad en la Integración de Datos:

- Dificultades en la consolidación de información para análisis y toma de decisiones.
- Necesidad de herramientas y técnicas especializadas para resolver incompatibilidades.

Garbage in, garbage out

1.2 Fuentes de datos no estructurados y semiestructurados

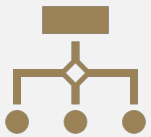
Objetivos:

- Comprender las características de los datos no estructurados y semiestructurados.
- Analizar ejemplos de datos provenientes de documentos, hojas de cálculo, grafos y textos.
- Familiarizarse con herramientas y técnicas para procesar y manejar estos tipos de datos.

Definición de datos estructurados, no estructurados y semiestructurados

Estructurados: Datos organizados en tablas con esquemas definidos (e.g., bases de datos relacionales).

No estructurados: Datos que no tienen un esquema predefinido y requieren técnicas especializadas para su análisis (e.g., imágenes, videos, texto plano).



Semiestructurados: Datos con cierta organización pero sin un esquema rígido (e.g., JSON, XML, bases vectoriales).

Tipos de datos y ejemplos prácticos

Documentos:

Archivos como PDF, Word, y presentaciones.

Hojas de cálculo:

Datos tabulares almacenados en Excel o Google Sheets.

Videos:

Archivos multimedia que contienen información visual y auditiva.

Caso práctico: Extracción de metadatos (duración, resolución) o generación de transcripciones con bibliotecas como FFmpeg o Google Speech-to-Text.

Tipos de datos y ejemplos prácticos

Grafos:

Representación de datos en nodos y aristas (e.g., redes sociales, grafos de conocimiento).

Textos:

Documentos en lenguaje natural como correos electrónicos, publicaciones en redes sociales o artículos.

Caso práctico: Procesamiento de texto con bibliotecas como NLTK o spaCy.

Tipos de datos y ejemplos prácticos

Documentos:

Archivos como PDF, Word, y presentaciones.

Caso práctico: Procesar contenido de documentos para extraer texto relevante usando OCR o bibliotecas como PyPDF2.

"Integración de Bases de Datos Vía WEB"

Blanca Abraham
FUNDACITE-Mérida Departamento de Computación
Mérida, Venezuela, 5101
Email: blanca@funmrd.gov.ve

José Aguilar, Francis Martínez
Universidad de los Andes, Facultad de Ingeniería
Mérida, Venezuela, 5101
Email: aguilar@ing.ula.ve , francismartinez@icnet.com.ve

Abstract

There are many organizations in the State of Mérida (Fundacite-Mérida, Zolccyt, etc.) that allow the registration of projects using automatic systems. Those systems handle insertion, modification, search and delete projects allowing the evaluation and promotion of those projects. Even when each system has a common main goal, that is, the registration of cultural, science and technology projects, they work independently, because of that, the idea of designing and implementing a Federated Database System was born, this system allows the web access to the autonomous systems and gives the users a global vision of the available regional information. This will give to each Institution the possibility of keeping the security and integrity of their data.

Keywords: 1. Database 2. Federated 3. System of Information 4. Multi Database Management System

Resumen

En el estado Mérida existen varias organizaciones (Fundacite-Mérida, Zolccyt, etc.) con sistemas automatizados de registro de proyectos, los cuales realizan las funciones básicas de inserción, modificación, consulta y eliminación de proyectos; y han sido elaborados con la intención de permitir el seguimiento, la evaluación y la promoción de los mismos. Considerando que cada uno de estos sistemas tienen como objeto el registro de proyectos orientados a áreas comunes (cultura, ciencia y tecnología), aunque con perfiles distintos, y que funcionan de manera aislada e independiente, nace la idea de diseñar e implementar un sistema de base de datos federadas que permita acceder vía web a estos sistemas de manera transparente, con el objeto de dar a los usuarios una visión global de la información disponible a escala regional. Esto permitiría a cada institución conservar la integridad, seguridad y propiedad de sus datos.

Palabras Claves: 1. Base de Datos 2. Federadas 3. Sistema de Información 4. Sistema Multibase de Datos

Tipos de datos y ejemplos prácticos

Hojas de cálculo:

Datos tabulares almacenados en Excel o Google Sheets.

Caso práctico: Manejo de datos desordenados o inconsistentes con Pandas.

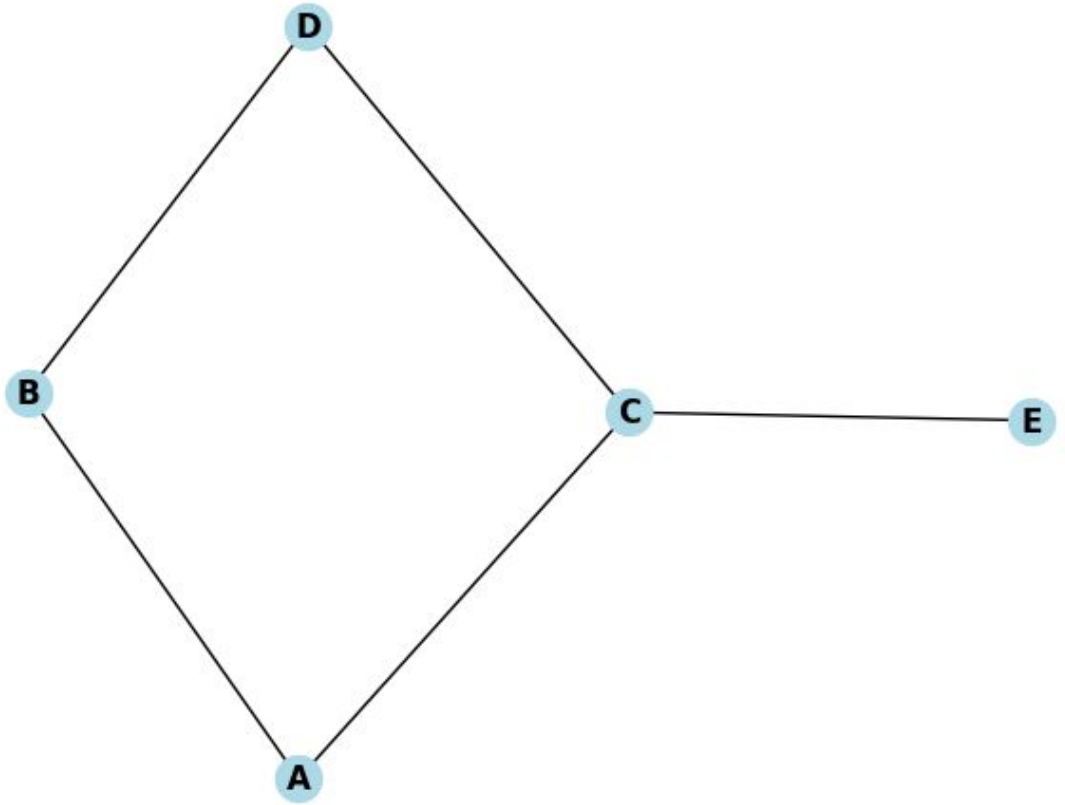
A	B	C	D	E
ID	**Nombre**	**Edad**	**Ciudad**	**Salario**
101	juan	23	México	$25,000
102	MARIA	45	Guadalajara	30,000
103	Luis	29	Monterrey	$45,000
	ana	34	tijuana	$NaN
105	Carlos		CDMX	50000
106		40	guadalajara	$60,000
107	Ricardo			40,000
108	MARTINA	27	Oaxaca	
109	sofia	33	Mérida	$55,500
110	Diego	41	Puebla	$70,000

Tipos de datos y ejemplos prácticos

Grafos:

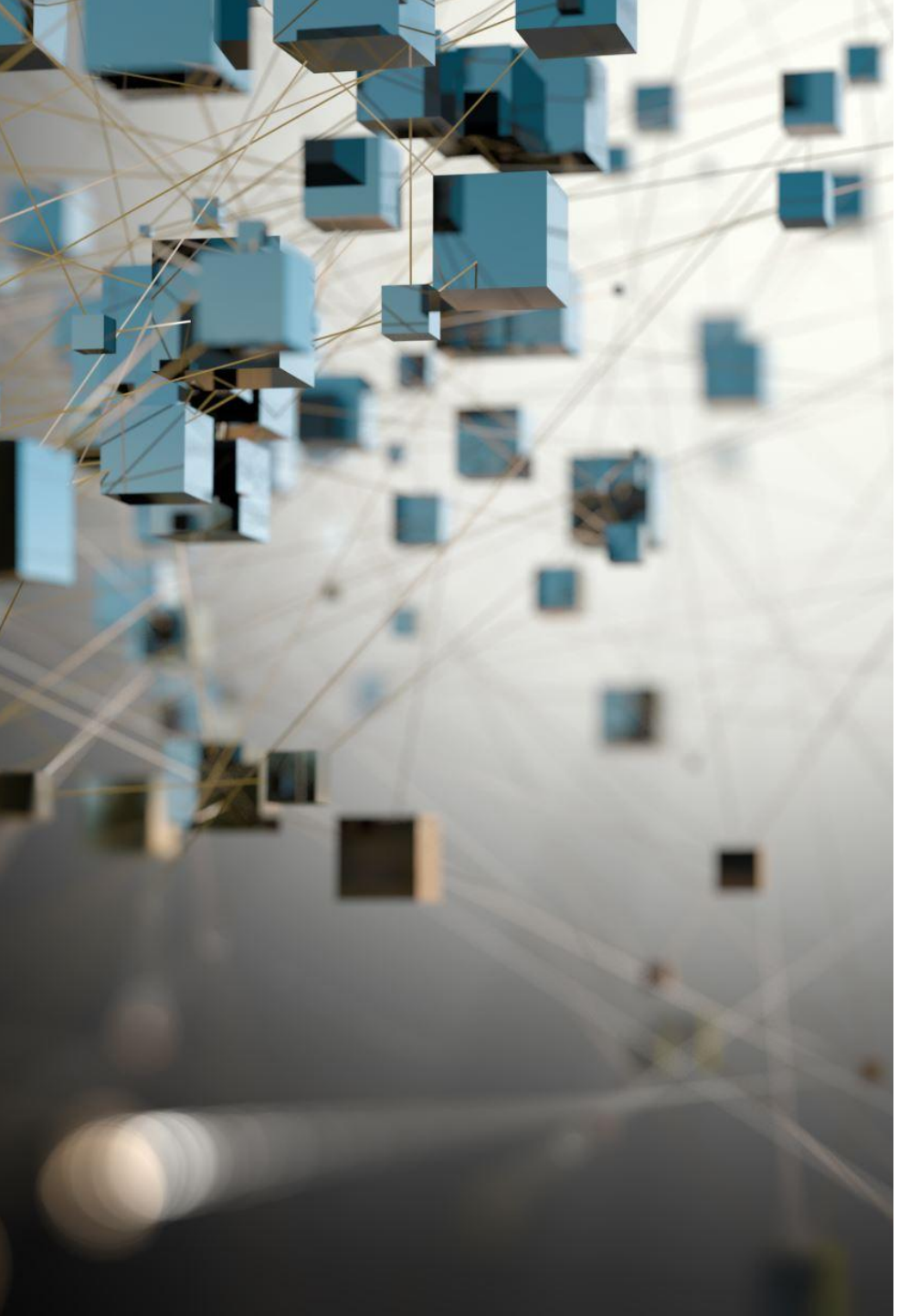
Representación de datos en nodos y aristas (e.g., redes sociales, grafos de conocimiento).

Caso práctico: Uso de NetworkX para analizar relaciones entre entidades.



Herramientas para el manejo de datos no estructurados y semiestructurados

- **Apache Hadoop:** Procesamiento distribuido para grandes volúmenes de datos estructurados y no estructurados.
- **MongoDB:** Base de datos NoSQL ideal para datos semiestructurados.
- **ElasticSearch:** Herramienta para búsqueda y análisis de grandes volúmenes de datos textuales.
- **Python:** Bibliotecas clave:
  - **PyPDF2:** Procesamiento de documentos PDF.
  - **Pandas, polars, pyspark:** Manipulación de datos tabulares y hojas de cálculo.
  - **FFmpeg:** Procesamiento de video.
  - **NLTK/spaCy:** Procesamiento de texto.
  - **networkx, igraph:** Gráficas.

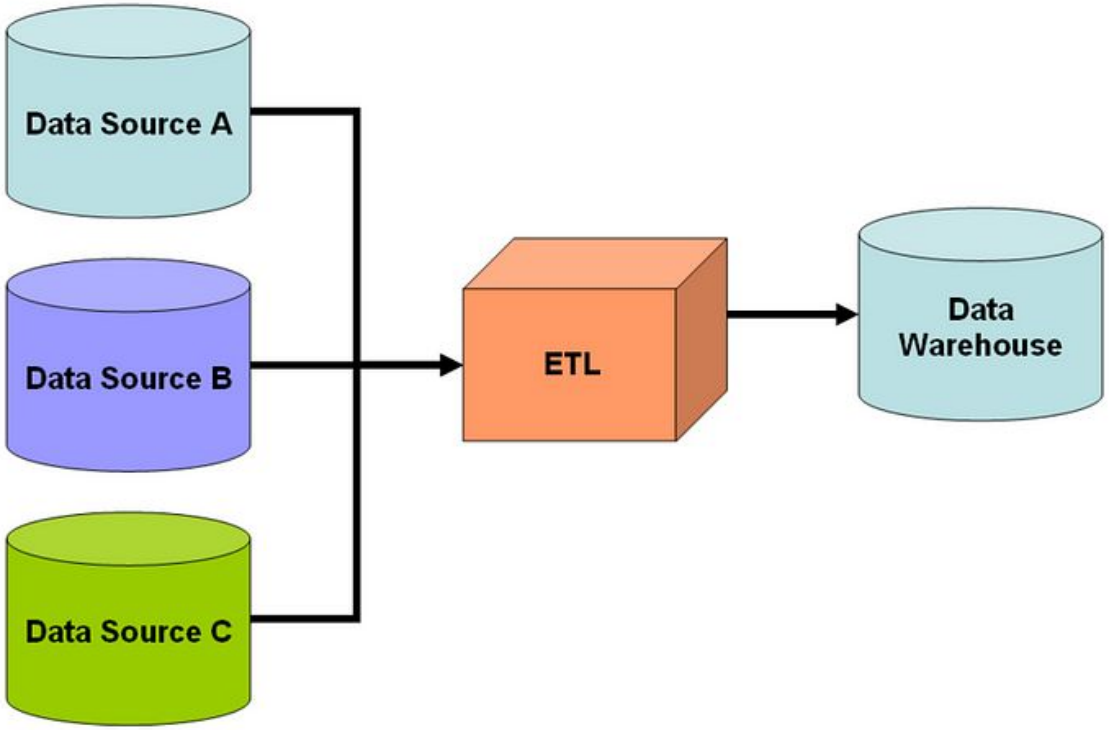


1.3 Integración de datos

Objetivos

- Comprender los diferentes enfoques de integración de datos y sus aplicaciones.
- Explorar el modelo canónico y su importancia en la estandarización de datos.
- Aprender sobre correspondencia y mapeo de esquemas en la integración de datos.

Métodos de Integración de Datos



Integración Materializada

- Los datos se almacenan en un repositorio centralizado (Data Warehouse).
- Se actualizan periódicamente mediante procesos ETL.
- Ejemplo: Consolidación de ventas de una empresa en una única base de datos.

Creación de bases a integrar

```sql
CREATE DATABASE IF NOT EXISTS demo;

CREATE OR REPLACE TABLE demo.clientes (
  id INT,
  nombre STRING
) USING DELTA;

INSERT INTO demo.clientes VALUES
  (1,'Ana'),
  (2,'Luis');

CREATE OR REPLACE TABLE demo.tx (
  tx INT,
  cliente_id INT,
  monto DOUBLE
) USING DELTA;

INSERT INTO demo.tx VALUES
  (101,1,250.0),
  (102,1,120.5),
  (103,2,999.9);
```

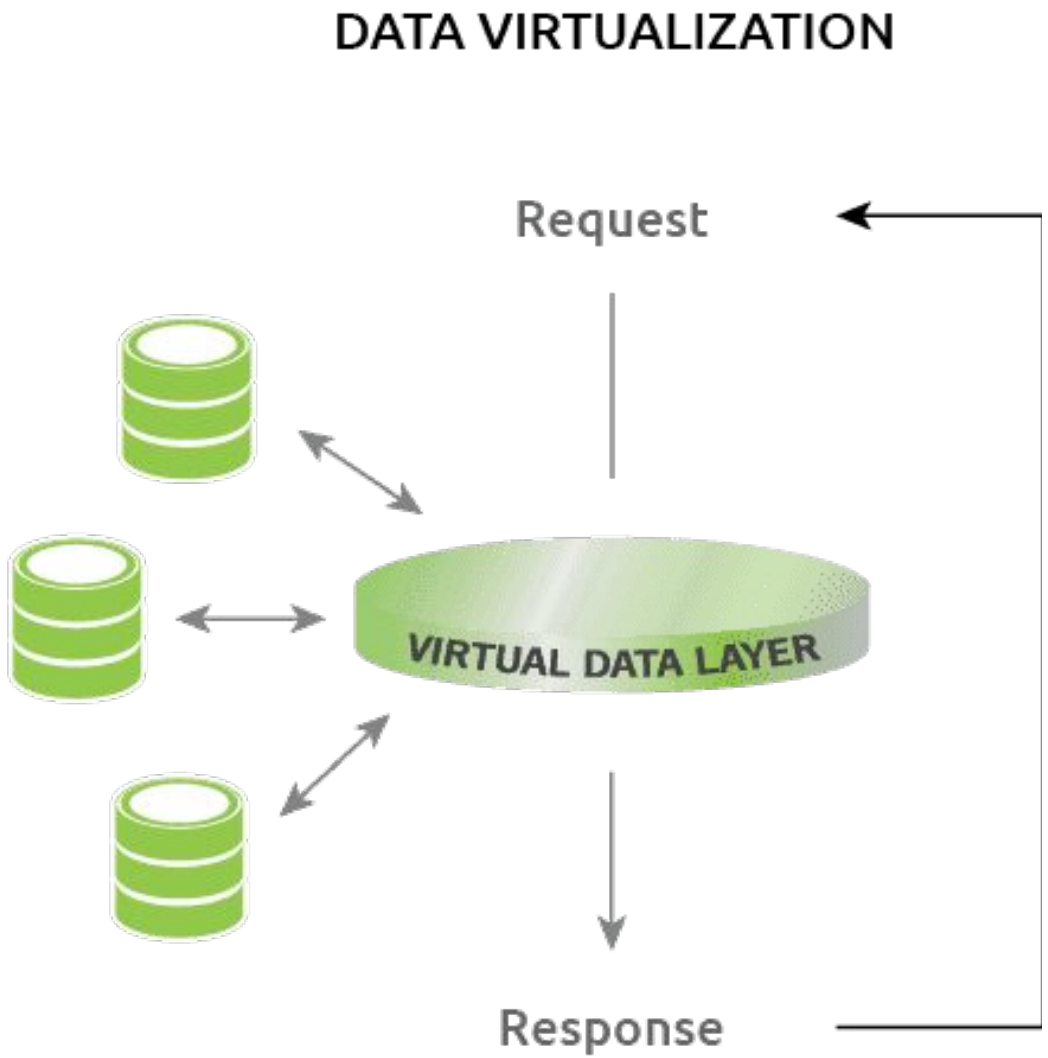
integración materializada en Databricks

```
1  CREATE OR REPLACE TABLE demo.tx_enriq
2  USING DELTA
3  AS
4  SELECT t.tx, c.nombre, t.monto
5  FROM demo.tx t
6  JOIN demo.clientes c
7  ON t.cliente_id = c.id;
8
9  select * from demo.tx_enriq;
```

Add parameter

Table +

	tx	nombre	monto
1	101	Ana	250
2	102	Ana	120.5
3	103	Luis	999.9



Métodos de Integración de Datos

Integración Virtual

- Accede a los datos en tiempo real sin moverlos.
- Usa vistas virtuales en lugar de almacenar copias de los datos.
- Ejemplo: Un portal bancario que consulta datos de diferentes sistemas sin duplicarlos.

integración virtual

```sql
CREATE OR REPLACE VIEW demo.vw_tx_enriq AS
SELECT t.tx, c.nombre, 2* t.monto as doble_monto
FROM demo.tx t
JOIN demo.clientes c
ON t.cliente_id = c.id;

select * from demo.vw_tx_enriq;

UPDATE demo.tx
SET monto = monto + 1000;

select * from demo.tx_enriq;
select * from demo.vw_tx_enriq;
```

Results 2 of 5 — Table

	tx	nombre	doble_monto
1	101	Ana	500
2	102	Ana	241
3	103	Luis	1999.8

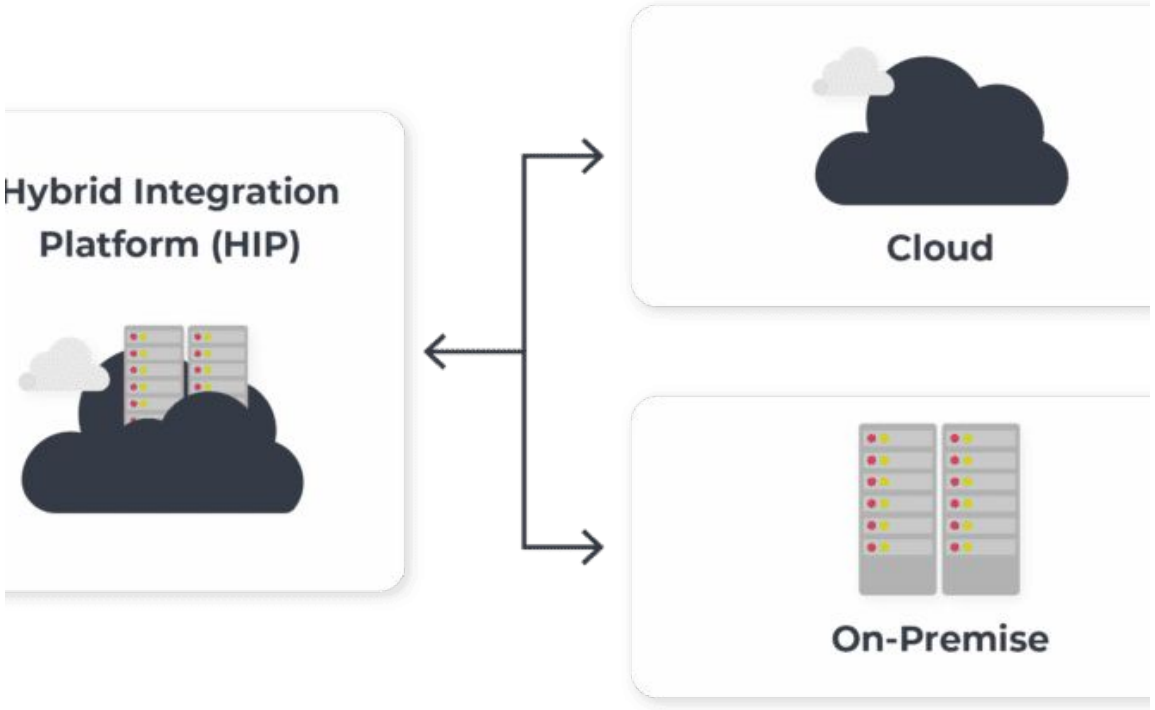
Results 4 of 5 — Table

	tx	nombre	monto	doble_monto
1	101	Ana	250	500
2	102	Ana	120.5	241
3	103	Luis	999.9	1999.8

Results 5 of 5 — Table

	tx	nombre	doble_monto
1	101	Ana	2500
2	102	Ana	2241
3	103	Luis	3999.8

Métodos de Integración de Datos

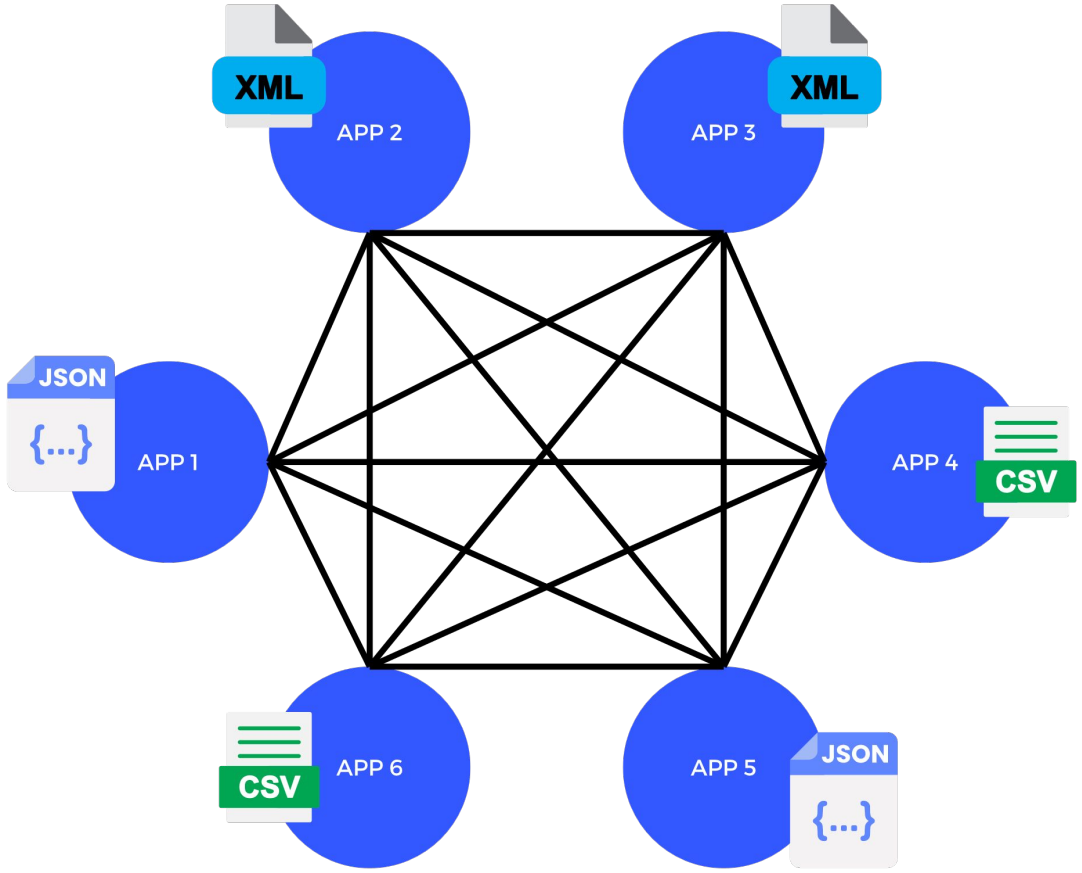


Integración Híbrida

- Combinación de integración materializada y virtual.
- Ejemplo: Un hospital que almacena historiales médicos en un Data Warehouse pero consulta en tiempo real exámenes de laboratorio.

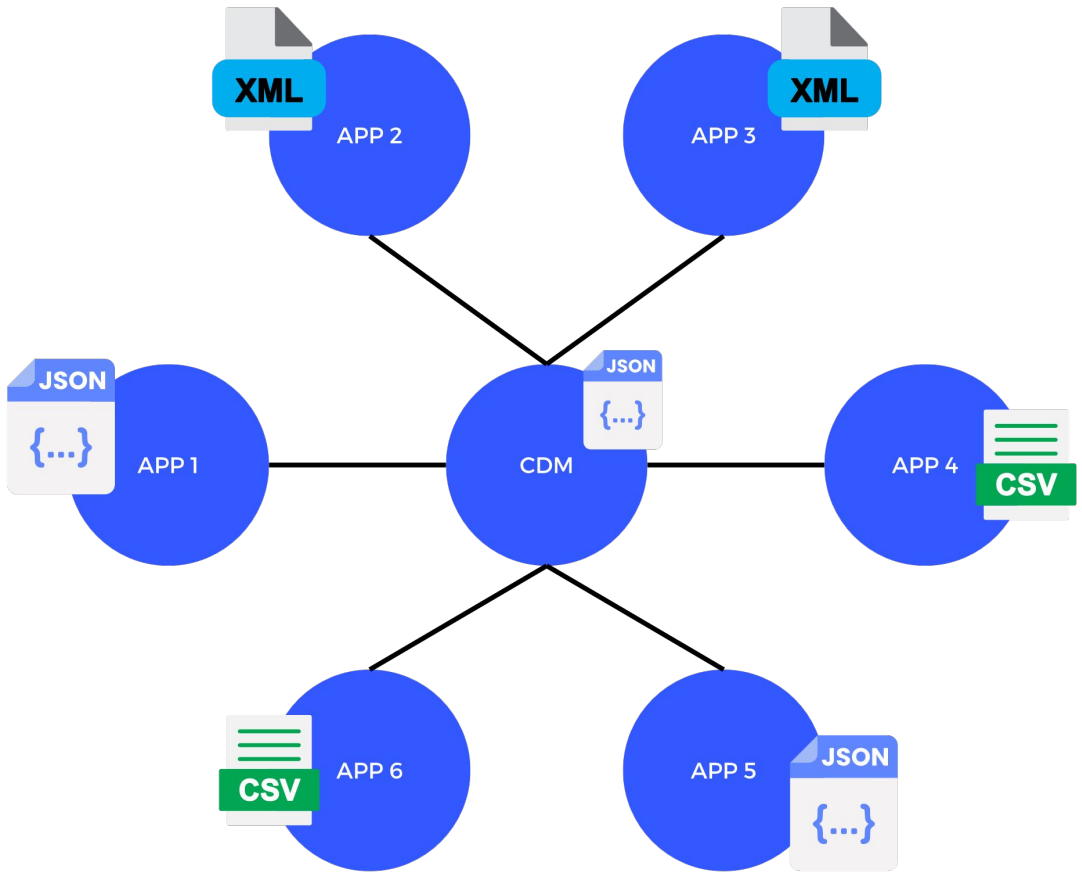
Modelo de datos canónico (CDM)

El modelo canónico en integración de datos es un enfoque que busca representar datos provenientes de múltiples fuentes heterogéneas en una estructura unificada e intermedia. Su propósito es proporcionar un esquema común que sirva como punto de referencia para la transformación y armonización de los datos antes de su integración.



Canonical Data Model (CDM)

En la arquitectura de aplicaciones, a menudo nos encontramos con el reto de conectar sistemas que almacenan datos en formatos incompatibles. La complejidad de establecer conexiones punto a punto entre cada aplicación es una fórmula para el caos, con un número de transformaciones que aumenta exponencialmente con cada nuevo sistema integrado.



Canonical Data Model (CDM)

La solución a esta escalada de complejidad es el Modelo de Datos Común (CDM). Este enfoque estandariza la forma en que los sistemas comparten datos, reduciendo drásticamente las transformaciones necesarias. Con el CDM, cada sistema solo necesita dos mapeos: uno hacia el CDM y otro de regreso a su formato nativo, simplificando la integración y haciéndola más escalable.

Ejemplo

Supongamos que una empresa tiene dos sistemas de bases de datos:

Un sistema relacional (MySQL) con una tabla de clientes:

ID	Nombre	Apellido	Teléfono
1	Juan	Pérez	555-1234
2	María	López	555-5678

Un sistema NoSQL (MongoDB) que almacena información similar en formato JSON:

```
{
  "customer_id": "A001",
  "full_name": "Juan Pérez",
  "contact": {"phone": "555-1234"}
}
```

Para integrar ambos en un modelo canónico, se puede definir un esquema intermedio como:

```
{
  "id": "string",
  "nombre_completo": "string",
  "telefono": "string"
}
```

Mapeo de Esquemas

- Identificar cómo los atributos de diferentes bases de datos están relacionados.
- Definir reglas para transformar datos de un esquema a otro.

Correspondencia y Mapeo de Esquemas

Ejemplo

- Una empresa fusiona sus bases de datos de clientes, pero cada sistema usa una estructura diferente:

Base de Datos A	Base de Datos B	Correspondencia
`ID_Cliente`	`User_ID`	✅ Equivalente
`Nombre_Cliente`	`ClienteNombre`	✅ Equivalente
`Telefono`	`Contacto`	✅ Equivalente
`Fecha_Registro`	`FechaAlta`	✅ Equivalente (pero diferente formato)
`Dirección`	*(No existe en B)*	❌ No hay correspondencia

Aquí podemos observar que hay una correspondencia total o parcial entre los esquemas, excepto por la columna Dirección, que no tiene equivalente en la Base B.

Extracción de Datos

Proceso de obtener datos de diferentes fuentes.

Ejemplo: Extraer datos de una API, de una red social como X, de una base de datos, etcétera.

Herramientas populares:

- SQL (para bases de datos relacionales).
- Python (bibliotecas como Pandas, Requests).

Transformación y Carga de Datos

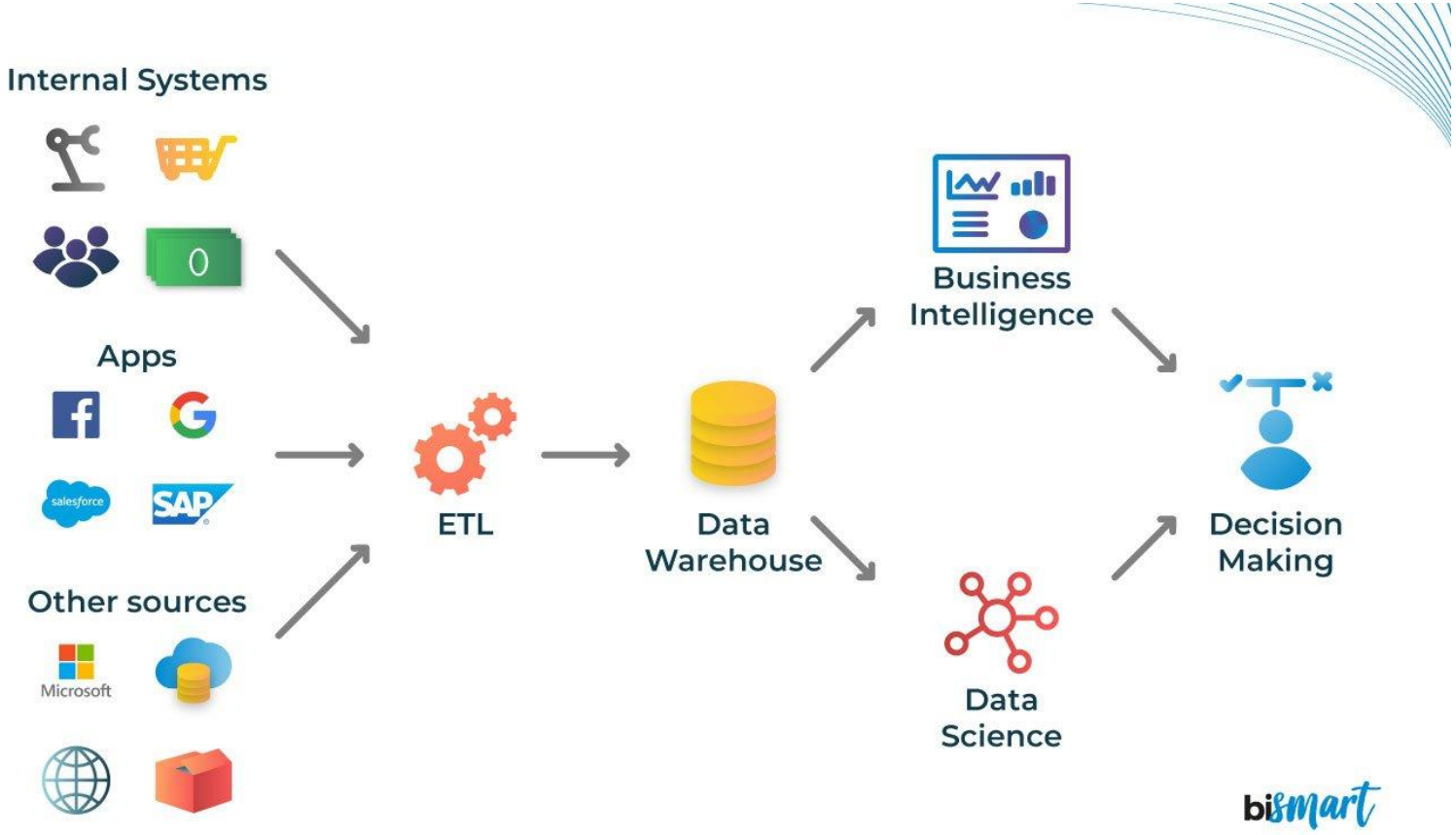
Transformación: Convertir, limpiar y estructurar los datos.

Carga: Almacenar los datos transformados en el sistema de destino.

Ejemplo:

- Extraer datos de un CSV, convertir fechas al formato correcto y cargarlos en una base de datos MySQL.

Extract, Transform and Load (ETL) (Extracción, Transformación y Carga)



Actividades

Discusión en Clase:

Analizaremos ejemplos reales donde la heterogeneidad de datos ha afectado procesos empresariales.

Debatiremos sobre posibles soluciones y enfoques para mitigar estos desafíos.

Integración de datos en la industria de la salud

En un hospital que utiliza múltiples sistemas de gestión para diferentes departamentos (admisiones, farmacia, laboratorio y facturación), la falta de integración de datos entre estos sistemas creó graves problemas. Los datos de los pacientes estaban distribuidos en diferentes formatos:

- El sistema de admisiones almacenaba datos en una base de datos relacional (SQL).
- El laboratorio usaba archivos en formato CSV para registrar resultados.
- La farmacia empleaba un sistema basado en XML para manejar inventarios y recetas.

Integración de datos en la industria de la salud

Problema:

- **Heterogeneidad Semántica:** Campos como "ID paciente" tenían diferentes definiciones en los sistemas: en uno era un número único y en otro un número combinado con una fecha.
- **Heterogeneidad Sintáctica:** Las estructuras de datos y los formatos dificultaban la interoperabilidad. Por ejemplo, los resultados de laboratorio eran exportados como tablas planas en CSV, pero el sistema de facturación requería un formato XML específico.

Integración de datos en la industria de la salud

Impacto:

- La falta de integración llevó a errores en los datos, como medicamentos prescritos incorrectamente debido a identificaciones de pacientes ambiguas.
- Procesos como la facturación médica y la emisión de reportes de resultados a los pacientes se retrasaron significativamente.
- El tiempo de respuesta para emergencias aumentó debido a la falta de un sistema centralizado para acceder al historial médico completo.

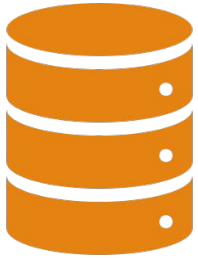
Propuestas de solución

Integración de datos en la industria de la salud

Solución Implementada: El hospital adoptó un enfoque híbrido de integración de datos:

1. **Materialización:** Un Data Warehouse consolidó la información crítica de todos los sistemas.
2. **Virtualización:** Para datos en tiempo real, se crearon vistas virtuales que transformaban automáticamente los formatos de entrada.
3. **Mapeo Semántico:** Se definieron reglas estándar para interpretar campos clave como "ID paciente."

Análisis



Estudio de Casos:

Revisaremos casos de estudio donde se implementaron estrategias exitosas de integración de datos heterogéneos.

Fusión de bancos y consolidación de datos

Dos grandes bancos internacionales, cada uno con sistemas de gestión de clientes (CRM) y transacciones financieras diferentes, se fusionaron. Los sistemas variaban:

- Banco A usaba un CRM basado en Microsoft Dynamics con datos almacenados en SQL Server.
- Banco B usaba un sistema propietario con datos en un almacén en la nube (Amazon Redshift).

Fusión de bancos y consolidación de datos

Problema:

- **Heterogeneidad Semántica:** Campos como "tipo de cliente" variaban ampliamente. En Banco A, se clasificaban como "retail" y "corporativo," mientras que Banco B usaba categorías como "individual" y "empresarial."
- **Heterogeneidad Sintáctica:** La estructura del CRM de Banco B no era compatible con el de Banco A. Por ejemplo, las transacciones se representaban como JSON anidado en Banco B, mientras que Banco A utilizaba tablas relacionales.

Fusión de bancos y consolidación de datos

Impacto:

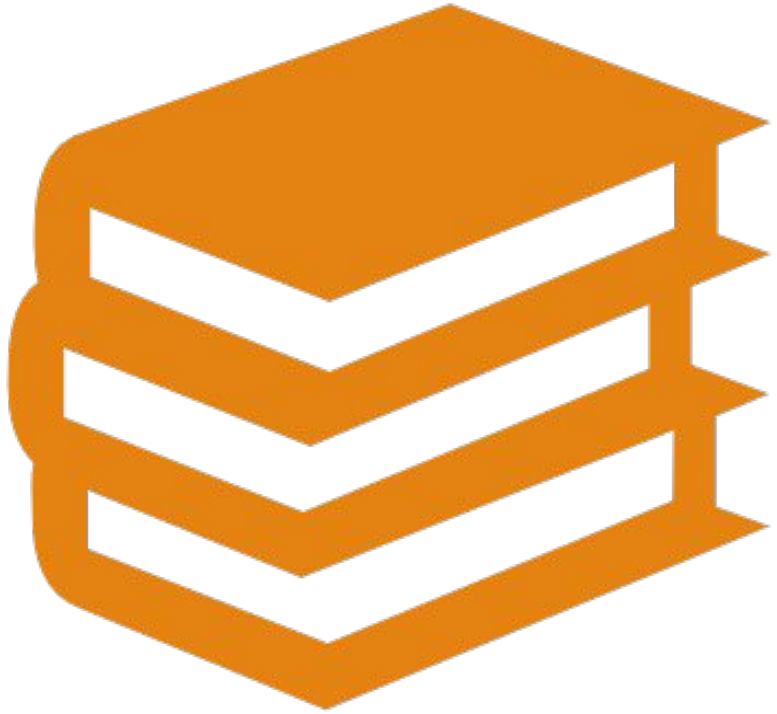
- **Duplicación de Clientes:** Clientes comunes a ambos bancos aparecían como entidades distintas, lo que causó confusión en las estrategias de marketing y riesgos en la gestión de créditos.
- **Desacuerdo en Reportes Financieros:** Diferencias en la forma de calcular el ingreso promedio por cliente afectaron los reportes consolidados, retrasando decisiones estratégicas clave.

Solución implementada

Fusión de bancos y consolidación de datos

Solución Implementada:

1. Se diseñó un modelo canónico para unificar las definiciones de datos entre los bancos.
2. Se implementaron procesos ETL para transformar los datos de ambos sistemas hacia un único formato estándar.
3. Un middleware de integración permitió acceder a los datos desde ambas plataformas mientras se completaba la transición al sistema consolidado.



Otros estudios de casos

Spread of contagious diseases

By integrating legacy data with social media data, health organizations can make better predictions about the spread of contagious diseases — and thus make more informed decisions to protect public health. Even a day or two saved in getting vaccines to the right location or implementing a quarantine can make a huge difference in limiting the extent of a health crisis.

Very recently, researchers at a renowned East-Coast university experimented with a new approach aimed at providing more immediate information about the current status of flu infections. Rather than relying on traditional data sources, the researchers used advanced algorithms to look at unstructured data from Twitter feeds. The researchers analyzed hundreds of thousands of tweets to determine locations where people had the flu. This social media data provided an up-to-the-minute and predictive look at where the flu was spreading at a particular point in time. Without integrating the new types of data from social media with the existing legacy data, it simply wouldn't be possible to gain this quick insight.

In the past, the data that's been available for tracking flu outbreaks has been traditional, legacy type data that only tells you what happened after the fact — and after the data has been collected and analyzed. In other words, you can see what has happened in various parts of the country, but because the information is at least several weeks old, it's not very useful for predicting where you'll need to send additional supplies of flu vaccine or where you'll need to ramp up staffing levels at clinics and hospitals in order to stem the epidemic.

Government

Governmental organizations aren't immune to the need to leverage data in new ways, either. Tight budgets mean that more efficiency is vital to providing services with the limited available resources. Data integration makes it possible for government departments to make the best use of both data and funding.

Consider a state transportation department with thousands of miles of roads and bridges. Its transportation system is recognized as one of the most accessible, efficient, and safe systems in the United States; however, the same could not be said of the IT infrastructure that supported the state department of transportation's (DOT's) financial, construction, maintenance, and traffic safety programs. Most of the agency's information systems — some dating to the late 1970s — were based on a legacy mainframe and a rather old-fashioned database, with a highly indexed, hierarchical data structure at odds with modern relational data systems. Using that legacy data was very difficult, and trying to combine it with modern data efficiently was almost impossible.

Streamlining the data integration process was vital in enabling the DOT to rapidly realize its primary objectives. Modern data integration tools made it possible to quickly combine the legacy data and the newer data sources into valuable information that would make sound decision making possible.

Referencias

- Batini, C., Lenzerini, M., & Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. ACM Computing Surveys, 18(4), 323-364.
- Doan, A., Halevy, A., & Ives, Z. (2012). Principles of Data Integration. Elsevier.
- Lenzerini, M. (2002). Data integration: A theoretical perspective. Proceedings of the ACM Symposium on Principles of Database Systems (PODS), 233-246.
- Rahm, E., & Do, H. H. (2000). Data Cleaning: Problems and Current Approaches. IEEE Data Engineering Bulletin, 23(4), 3-13.
- Halevy, A., Rajaraman, A., & Ordille, J. (2006). Data integration: The teenage years. Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), 9-16.
- Bernstein, P. A., & Rahm, E. (2001). Data warehouse systems: architecture and methods. ACM SIGMOD Record, 30(2), 6-15.
- Gagnon, M. (2007). Ontology-based integration of data sources. Information Fusion, 8(1), 33-46.